UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Carlos Augusto Berlitz

Plataforma de Testes para Caracterização Elétrica de Diodos Schottky Integrados

Porto Alegre

2017

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Carlos Augusto Berlitz

Plataforma de Testes para Caracterização Elétrica de Diodos Schottky Integrados

Projeto de Diplomação apresentado ao Departamento de Engenharia Elétrica da Escola de Engenharia da Universidade Federal do Rio Grande do Sul, como requisito parcial para Graduação em Engenharia Elétrica

Orientador: Prof. Dr. Hamilton Duarte Klimach

Porto Alegre
2017

Berlitz, Carlos Augusto
Plataforma de Testes para Caracterização Elétrica de Diodos Schottky Integrados/ Berlitz, Carlos Augusto. – Porto Alegre, 2017-
55 p.

Orientador: Prof. Dr. Hamilton Duarte Klimach

Trabalho de Conclusão de Curso – Universidade Federal do Rio Grande do Sul, Escola de Engenharia, Curso de Engenharia Elétrica,
Porto Alegre, BR-RS, 2017. 2017.

1. Microeletrônica. 2. Diodo Schottky. 3. Referência de tensão. 4. Plataforma de testes. 5. Parametrização. 6. Variabilidade. I. Klimach, Hamilton, orientador. II. Universidade Federal do Rio Grande do Sul. III. Escola de Engenharia. IV. Departamento de Engenharia Elétrica. V. Título.

CARLOS AUGUSTO BERLITZ

Plataforma de Testes para Caracterização Elétrica de Diodos Schottky Integrados

Projeto de Diplomação apresentado ao Departamento de Engenharia Elétrica da Escola de Engenharia da Universidade Federal do Rio Grande do Sul, como requisito parcial para Graduação em Engenharia Elétrica

Trabalho aprovado. Porto Alegre, dia 25 de julho de 2017:

Prof. Dr. Hamilton Duarte Klimach
Orientador - UFRGS

Prof. Dr. Ály Ferreira Flores Filho
Chefe do Departamento de Engenharia Elétrica (DELET) - UFRGS

Banca Examinadora:

Prof. Dr. Luiz Fernando Ferreira
Professor do Departamento de Engenharia Elétrica - UFRGS

MSc. Israel Sperotto
Doutorando no PPG em Microeletrônica - UFRGS

Eng. Alonso Aymone de Almeida Schmidt
Instrutor no NSCAD/UFRGS

Dedico este trabalho aos meus amigos que estiveram presentes durante essa minha jornada da graduação, à minha família que habilitou essa jornada e especialmente à minha namorada Dayane que foi acima de tudo uma companheira exemplar no decorrer desse meu caminho.

Agradecimentos

Agradeço primeiramente a minha família por me dar a base necessária para chegar até aqui me fornecendo apoio e incentivo sempre que necessário.

Agradeço principalmente à minha companheira Dayane Leal que sempre esteve presente do meu lado, me dando suporte e me mantendo no curso correto e motivado para encarar os desafios que se apresentaram. Aproveito para agradecer à sua família, em especial à Maregilda pelo auxílio em diversos momentos.

Também agradeço à Rosângela, à Ana e ao Marlon pelos cafés e os bons momentos na Procuradoria que auxiliaram a minha jornada.

Trago meu agradecimento ao grupo AMS do programa PGMICRO pelo apoio e auxílio no desenvolvimento desse projeto e pela expertise proporcionada. Além do meu agradecimento à TU Dresden, mais especifiamente ao curso *Nanoelectronic Systems* pela oportunidade e me mostrar a paixão pela microeletrônica.

Agradeço ao professor Hamilton Klimach pelos ensinamentos não somente específicos à área de estudo, pela disponibilidade antes e durante a execução desse trabalho e pela dedicação ímpar.

Não posso deixar de agradecer aos meus amigos que proporcionaram momentos memoráveis e necessários de descontração durante a árdua jornada desse curso de graduação. Não menos importante aos meus colegas e professores, pelas horas de estudo, trabalho e pesquisa, além das experiências alheias ao núcleo acadêmico. Em especial agradeço aos meus amigos Felipe Kalinski, Fernando Ferreira e Rafael Paz que além do grande companheirismo nas atividades acadêmicas se mostraram valorosas pessoas e amigos de verdade sem os quais eu não teria chegado ao fim dessa jornada de maneira satisfatória.

"Tudo o que temos de decidir é o que fazer com o tempo que nos é dado."

(J. R. R. Tolkien)

Resumo

A redução da tensão de alimentação dos dispositivos eletrônicos traz a necessidade do estudo de novas topologias para referências de tensão. Essas topologias buscam o uso de dispositivos normalmente alheios a esse tipo de aplicações, como diodos Schottky, encontrados usualmente em circuitos de proteção para circuitos integrados (CIs). Para esse uso é mandatório que os parâmetros de variabilidade sejam bem conhecidos e, devido a ausência de tais dados por parte da indústria, surge a necessidade do desenvolvimento de estruturas para caracterização e obtenção dos parâmetros relevantes para a aplicação. Sistemas de caracterização de dispositivos semicondutores tem em seu cerne analisadores de parâmetros, isso traz a necessidade do desenvolvimento de estruturas de integração entre estes e outros equipamentos, como câmaras térmicas para conformar o ambiente e circuitos para isolação galvânica entre o dispositivo a ser medido e o computador. Também é necessária uma estrutura de controle para automatizar e deixar mais modular o sistema de caracterização possibilitando reuso futuro em aplicações distintas. A estrutura modular desenvolvida habilita a caracterização automatizada da matriz de diodos Schottky e provê diversas possibilidades de aproveitamento futuro dessa tecnologia.

Palavras-chave: Microeletrônica. Diodo Schottky. Referência de Tensão. Plataforma de testes. Parametrização. Variabilidade.

Abstract

The reduction of the supply voltage of electronic circuits brings the need to study new topologies for voltage reference circuits. These topologies seek the use of devices normally alien to this kind of application, like Schottky diodes, which are usually found in circuit protection for integrated circuits (ICs). Aiming at this use it is mandatory for the variability parameters to be well known and, due to the lack of this data from the industry side, arise the need to develop structures for characterizing and obtaining the application relevant parameters. Semiconductor characterization systems have as its core a parameter analyzer, this brings the need to develop integration structures to act between this and other equipments, like thermal chambers to conform the ambient and isolation circuits for galvanic isolation between the device under test and the computer. There is also the need of a control structure to automate and transform the characterization system in something more modular allowing the reuse in future different applications. The modular developed structure enables automated characterizaton of a Schottky diode matrix and offers many future uses for the aplication of this technology.

Keywords: Microelectronics. Schottky Diode. Voltage Reference. Testbench. Parameterization. Variability.

Lista de ilustrações

Lista de abreviaturas e siglas

ABNT	*Associação Brasileira de Normas Técnicas*
IEEE	*Instituto de Engenheiros Eletricistas e Eletrônicos*
IEC	*Comissão Eletrônica Internacional*
PGMICRO	*Programa de Pós-Graduação em Microeletrônica*
CI	*Circuito Integrado*
V	*Volt*
MOS	*Metal Oxide Semiconductor*
AMS	*Analog and Mixed Signal*
Vdd	*Tensão de Alimentação*
D/A	*Digital/Analógico*
IoT	*Internet of Things (em inglês: Internet das Coisas)*
N	*Semicondutor tipo N*
P	*Semicondutor tipo P*
USB	*Universal Serial Bus (em inglês: Porta Serial Universal)*
GPIB	*General Purpose Interface Bus (em inglês: Porta de Interface de Propósito Geral)*

Sumário

1 Introdução

Com a crescente demanda por conectividade, em grande parte pela tendência estabelecida pela Internet das Coisas (em inglês: *Internet of Things*), aumenta também o uso de dispositivos móveis, sensores e atuadores remotos. Soma-se a isso as aplicações que tem a mobilidade em seu cerne e vemos que há uma necessidade de se diminuir o consumo de energia elétrica dos dispositivos. Apesar de esforços para substituição das baterias por outras tecnologias como *energy harvesting* (captação de energia do ambiente) mesmo essas necessitam que o consumo do circuito não seja elevado.

Como visto em (SZE; NG, 2006), em dispositivos digitais o consumo se baseia principalmente no consumo dinâmico e este possui uma dependência quadrática com a tensão de alimentação, uma alternativa para obter a necessária diminuição de consumo é a redução da tensão de alimentação. Visando tal redução novas topologias de circuitos eletrônicos devem ser estudadas e desenvolvidas. Entre as topologias usadas para referência de tensão, a referência de tensão por *bandgap* figura entre as mais populares, no entanto a sua abordagem fazendo uso de diodos de junção PN encontra limitações quanto ao valor que pode ser alcançado. Para valores de tensão abaixo de $1V$ a abordagem tradicional sofre alterações, uma das alternativas surge ao substituir o diodo de junção PN por um diodo Schottky como vemos em (V. HAMILTON KLIMACH, 2016).

Nos processos de fabricação em massa há variações entre as diversas unidades produzidas, sendo fruto do fato dos processos de fabricação não serem absolutamente precisos. Na produção de circuitos integrados não é diferente, conforme apresentado em (KLIMACH, 2008) há variações físicas decorrentes das diversas etapas do processo de fabricação de circuitos integrados, aqui chamadas de variabilidade de fabricação. Essa variabilidade de fabricação acarreta em variações de comportamento dos CIs produzidos.

Diodos Schottky encontram sua aplicação mais popular em circuitos de proteção de CIs, onde não há grande necessidade de uma caracterização aprofundada de seus parâmetros. No entanto, para o uso em circuitos de referência de tensão é necessário o conhecimento de seus parâmetros de variabilidade dentro do processo de fabricação, visto que os mesmos influenciam largamente o comportamento do circuito projetado. Para circuitos de referências de tensão é importante ressaltar o impacto que a variabilidade do processo de fabricação causa sobre o mesmo, além das influências causadas por temperatura, ruídos de corrente, alterações na tensão de alimentação, entre outras.

A obtenção desses parâmetros de variabilidade é feita de maneira estatística e, como apresentado em (SCHMID, 2014), o número de amostras necessárias para se tirar conclusões com elevada confiabilidade é considerável. Com esse elevado número de amostras (medições)

supõe-se que a incerteza comportamental seja aleatória e estima-se os parâmetros de variabilidade pela média e desvio-padrão.

Para a obtenção desses parâmetros foi proposto e posto em prática o trabalho apresentado em (PEDRINI THALES STEDILE, 2016) onde foram confeccionados 4 grupos, cada um contendo 100 diodos. Cada um desses 4 grupos foi confeccionado com dimensões distintas. Para a obtenção dos parâmetros de variabilidade do processo de fabricação serão caracterizados os diodos produzidos. Essa caracterização deve ser feita de forma automatizada fazendo uso de uma plataforma de testes e caracterização especialmente desenvolvida.

A plataforma desenvolvida consiste de um analisador de parâmetros, uma câmara de testes, um computador, um circuito de isolação e o próprio dispositivo a ser testado, tal sistema tem como base o apresentado em (KLIMACH, 2008). Foi desenvolvido também um protocolo de medição para que os resultados sejam confiáveis e comparáveis. Com a modularidade da plataforma desenvolvida, ela habilita um grande reuso da mesma em futuros projetos e mesmo projetos atuais que necessitam de caracterização e testes.

1.1 Motivação

A diminuição na tensão de alimentação dos circuitos integrados traz a necesidade de estudos de novas topologias de referências de tensão. Essa diminuição busca uma redução no consumo dos dispositivos e circuitos. A diminuição da tensão de alimentação se baseia na a dependência quadrática do consumo com a tensão de alimentação como visto em (SZE; NG, 2006).

Novas topologias para circuitos de referência de tensão podem se basear no uso de dispositivos antes alheios a essas aplicações como visto em (V. HAMILTON KLIMACH, 2016). Nesse trabalho vemos o uso de diodos Schottky em um circuito de referência de tensão por *bandgap* no lugar do mais popular diodo de junção PN. Essa é uma das mudanças possíveis para se alcançar referências de tensão para valores inferiores a $1V$. Para o uso de diodos Schottky nesse tipo de aplicação se faz necessário o uso de parâmetros de variabilidade mais precisos durante a fase de projeto visto que a influência da variabilidade do processo de fabricação é preponderante juntamente com a baixa sensibilidade a temperatura, sem esquecer da sensibilidade à tensão de alimentação, ruído de corrente, entre outros.

Visto que os parâmetros do processo de fabricação que esses não são oferecidos pela indústria para esses dispositivos, foi desenvolvido o projeto descrito em (PEDRINI THALES STEDILE, 2016) onde se projetou uma matriz com 400 diodos Schottky de diferentes tamanhos agrupados em grupos de 100. Com ele é visada a obtenção de tais parâmetros do processo de fabricação. Para que esse circuito seja caracterizado e os parâmetros de variabilidade de fabricação do diodo Schottky sejam conhecidos é necessária a caracterização dos diodos fabricados alcançada de forma automatizada com o uso de uma plataforma de testes e caracterização

especialmente desenvolvida. Essa plataforma de testes necessita atender diversas restrições e requisitos propostos pelo circuito da matriz de diodos para que os resultados sejam válidos para a aplicação desejada.

1.2 Justificativa

A caracterização dos diodos Schottky projetados em (PEDRINI THALES STEDILE, 2016) traz a necessidade de um método automatizado devido ao seu número elevado de dispositivos e a necessidade de caracterização individual e confiável dos mesmos. Alia-se a isso as necessidades do grupo de AMS do programa PGMICRO de uma estrutura de testes automatizada para caracterização e testes dos projetos por eles desenvolvidos e fabricados, mostrando a relevância contínua e futura para a plataforma. Tal plataforma desenvolvida de forma modular possibilita também futuros desenvolvimentos para usos mais específicos habilitando aplicações em áreas hoje deixadas em segundo plano pelos grupos de pesquisa da universidade.

1.3 Objetivo

Esse trabalho visa primeiramente o estudo dos equipamentos a serem usados para o desenvolvimento da estrutura de testes seguida pela implementação de tal estrutura, de maneira modular e documentada.Além disso serão realizadas medidas para um grupo de diodos Schottky visando a validação da estrutura.

1.4 Organização

Esse trabalho se organiza da seguinte maneira. Após a introdução o próximo capítulo fornece a fundamentação teórica em que esse trabalho se baseia, os dispositivos a serem medidos e equipamentos usados no sistema de medição. Em seguida temos um capítulo onde são apresentados e explicados os softwares desenvolvidos para essa estrutura. No capítulo 4 é então descrito o hardware desenvolvido e utilizado seguido pelo protocolo de medição no capítulo 5. Temos no capítulo 6 a apresentação das conclusões desse trabalho e por fim sendo feitas propostas para futuros trabalhos.

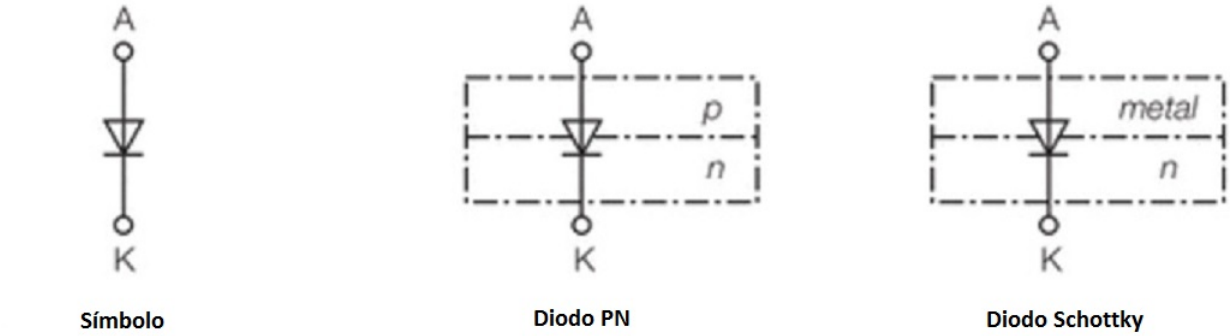
2 FUNDAMENTAÇÃO TEÓRICA

Para o sistema de medidas desenvolvido, os pontos críticos englobam desde o dispositivo principal a ser medido, o diodo Schottky, até os equipamentos a serem utilizados. É interessante ressaltar os requisitos de alguns dos equipamentos, como a necessidade do analisador de parâmetros possuir a função *force-sense*. Nesse capítulo o foco se dá em apresentar os fundamentos teóricos utilizados para o desenvolvimento deste trabalho, iniciando com uma descrição de diodos Schottky, incluindo uma breve descrição de diodos, passando pelos equipamentos a serem utilizados, onde são descritos suas características e propriedades.

2.1 Diodos

Segundo (TIETZE; SCHENK; GAMM, 2008) o diodo é um dispositivo semicondutor com dois terminais, um conhecido como ânodo (A) e outro como cátodo (K). Diodos consistem de uma junção PN ou de uma junção metal-n, sendo conhecidos como diodo de junção PN e diodo Schottky, respectivamente. A Figura 1 mostra o primeiro, à esquerda, o símbolo de um diodo de junção, enquanto ao centro e à direita a diferença de construção entre, respectivamente, o diodo PN e o diodo Schottky. Utilizamos a letra A para representa o ânodo e a letra K o cátodo dos diodos. O diodo de junção PN é feito utilizando-se silício e normalmente chamado somente de diodo, enquanto as outras variantes possuem algum termo extra.

Figura 1 – Símbolo e construção dos diodos.

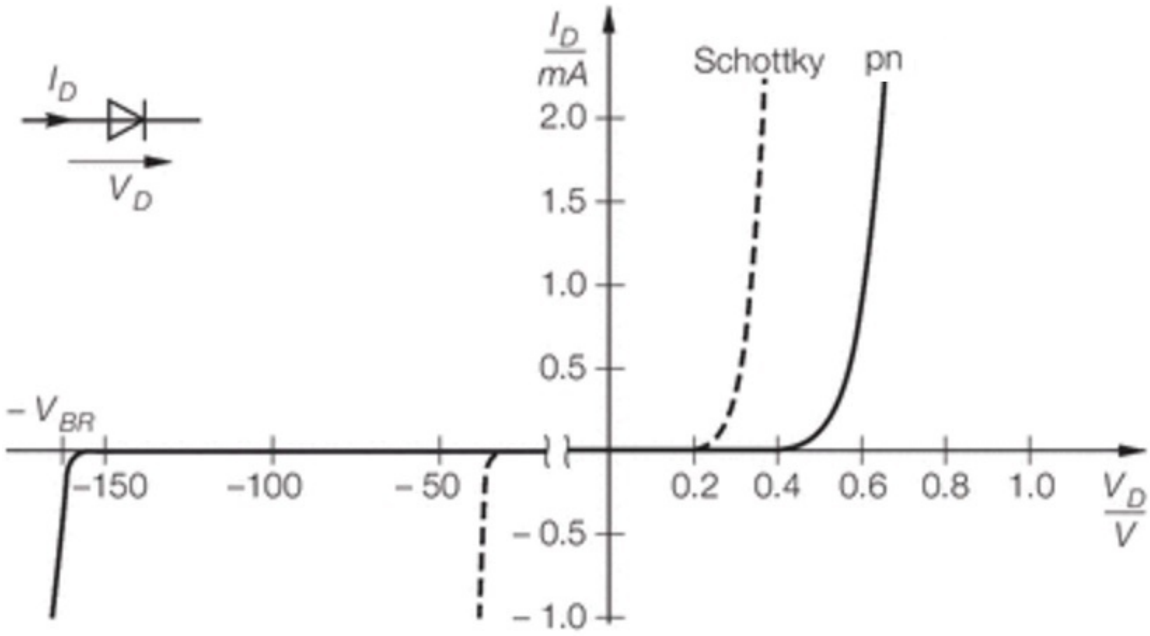


Fonte: (TIETZE; SCHENK; GAMM, 2008)

A performance de um diodo pode ser claramente observada pela sua curva característica exposta na Figura 2, onde vemos lado a lado as curvas para um diodo Schottky e para um diodo de silício de junção PN. A curva mostra a relação entre corrente e tensão enquanto os outros parâmetros são mantidos constantes, ou mudam de forma extremamente lenta. No eixo horizontal vemos a tensão do diodo dada em Volts, enquanto no eixo vertical vemos a corrente no mesmo

dada em mA. A curva tracejada representa o diodo Schottky, com uma tensão de condução mais baixa, expressa pela maior proximidade da mesma com a origem e o eixo vertical. A curva sólida representa o diodo de junção PN tendo seus valores de tensão para condução superiores aos do diodo Schottky sendo a condução direta, à direita do eixo vertical, aproximadamente o dobro do diodo Schottky. À esquerda vemos o que é chamado de condução reversa onde temos também $-V_{BR}$, sendo V_{BR} a tensão de ruptura. A tensão de ruptura representa o valor de tensão onde começa a ocorrer a condução reversa para um determinado diodo e, se a corrente não for limitada, pode trazer danos ao dispositivo.

Figura 2 – Curva corrente por tensão de diodos.



Fonte: (TIETZE; SCHENK; GAMM, 2008)

A equação apresentada em 2.1, também conhecida como equação de Shockley para o diodo, descreve matematicamente o comportamento de um diodo real, onde vemos que a corrente do diodo I_D tem uma dependência logarítmica com a tensão do diodo V_D. Sabendo que I_S é a corrente de saturação em polarização reversa, n é o coeficiente de emissão, que define a não idealidade do diodo e V_T é a tensão térmica definida pela Equação 2.2. Nesta, k é a constante de Boltzmann, T é a temperatura absoluta e q a carga do elétron.

$$I_D = I_S \cdot \left(e^{\frac{V_D}{n \cdot V_T}} - 1 \right) \tag{2.1}$$

$$V_T = \frac{k \cdot T}{q} \tag{2.2}$$

2.1.1 Diodo Schottky

O diodo Schottky, que recebeu seu nome em homenagem ao físico alemão Walter H. Schottky, se diferencia do diodo de junção PN na contrução pois sua região p é substituída por uma região de metal. Essa alteração faz com que sua tensão de condução direta seja inferior e sua resposta dinâmica seja mais rápida quando comparado com o diodo de silício. Enquanto para o diodo de silício a queda de tensão típica fica entre 500 e 700 mV, para o diodo Schottky a mesma fica entre 150 e 450 mV (PEDRINI THALES STEDILE, 2016). Vemos na Figura 2 a diferença dessa tensão.

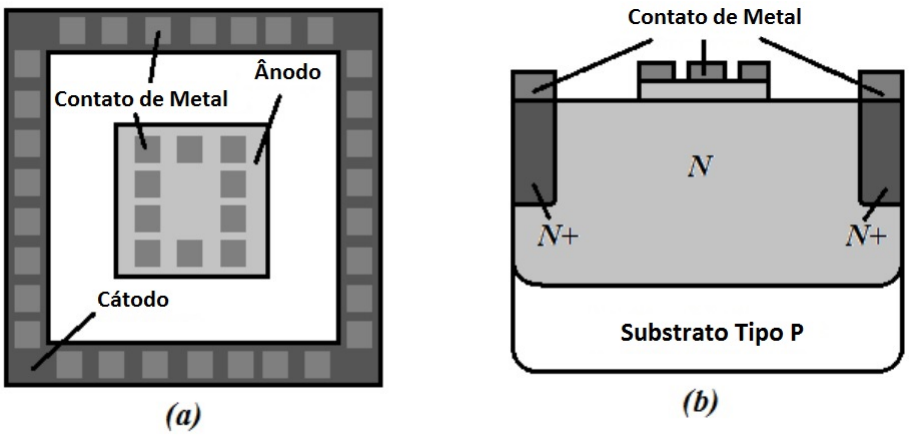
Como visto em (TIETZE; SCHENK; GAMM, 2008), embora o diodo Schottky seja criado através de uma juncão n-metal, é importante ressaltar que nem todas as juncões entre metais e semicondutores dão origem a um diodo Schottky, para que o mesmo seja criado é necessário que o semicondutor seja levemente dopado. A juncão entre o metal e o semicondutor criam uma barreira Schottky, no lugar da barreira semicondutor-semicondutor criada nos diodos de junção PN. Embora seja possível desenvolver uma barreira Schottky tanto com dopagem do tipo n, quanto do tipo p, tipicamente semicondutores do tipo n são escolhidos devido a tensão direta ser consideravelmente menor.

Os metais que são tipicamente utilizados são diferentes do metal usado para as conexões do circuito integrado, como platina, cromo e tungstênio, enquanto o semicondutor normalmente é silício. O metal age como o ânodo e o semicondutor como o cátodo.

O comportamento do diodo Schottky pode ser descrito por 2.3, onde J é a densidade de corrente, ϕ_{Bn} é o potencial da barreira, que depende dos detalhes construtivos, T é a temperatura absoluta, V_a é a tensão aplicada, q a carga do eletron, k a constante de Boltzmann e A é a constante de Richardson, que depende da mobilidade dos eletrons (que varia com a dopagem do semicondutor).

$$J = \left[A \cdot T^2 \cdot e^{\frac{-q \cdot \phi_{Bn}}{k \cdot T}}\right] \cdot \left[e^{\frac{q \cdot V_a}{k \cdot T}} - 1\right] \tag{2.3}$$

Figura 3 – Estrutura do diodo Schotkky para o processo IBM 130 nm.



A estrutura física do diodo Schottky, vista na Figura 3 para o processo IBM 130 nm, pode ser feita ao colocarmos um contato de metal sobre uma região semicondutora levemente dopada do tipo n, normalmente a camada do poço n, formando o lado do ânodo. Como no lado do cátodo não é desejado haver uma nova junção esses contatos de metal são posicionados sobre uma camada fortemente dopada do tipo n. Vemos a vista superior (a) e lateral (b) do processo.

2.2 Variabilidade de Fabricação

Conforme (KLIMACH, 2008) as variações de comportamento de dispositivos produzidos pela tecnologia MOS são decorrências de variações físicas entre os dispositivos nas várias etapas do processo de fabricação. Definimos que para esse trabalho essas variações do processo de fabricação são definidas pela variabilidade de fabricação.

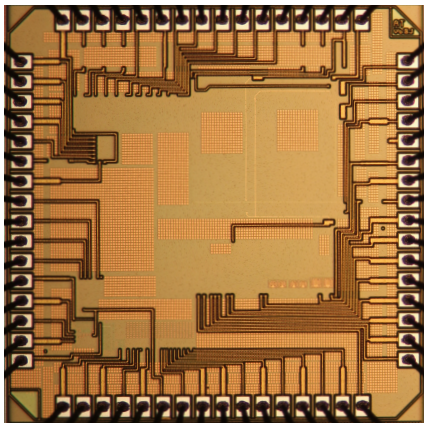
Nas lâminas em tecnologia MOS as principais variações físicas observadas são:

- flutuações na concentração e no perfil de dopantes no substrato e nas camadas eletricamente ativas;
- variações e flutuações na espessura e na qualidade das diversas camadas crescidas ou depositadas sobre o substrato;
- variações nas dimensões laterais das formas geométricas das camadas depositadas ou decapadas;
- rugosidade de borda das formas geométricas das camadas depositadas ou decapadas;
- flutuações na qualidade do contato elétrico entre camadas condutoras diferentes;
- tensões mecânicas permanentes na superfície do substrato.

2.3 Chip de teste

O chip fabricado em (PEDRINI THALES STEDILE, 2016) usa um encapsulamento PLCC de 68 pinos, vemos o chip na Figura 4. O chip é constituído de 400 diodos divididos em quatro grupos de 100, cada um contendo apenas diodos de uma geometria. As geometrias escolhidas foram primeiramente 2 μm x 2 μm, a segunda de 5 μm x 5 μm, seguida por 10 μm x 10 μm, enquanto a última também conta com diodos de geometria 10 μm x 10 μm, porém com topologia *fingered*.

Figura 4 – Chip fabricado.



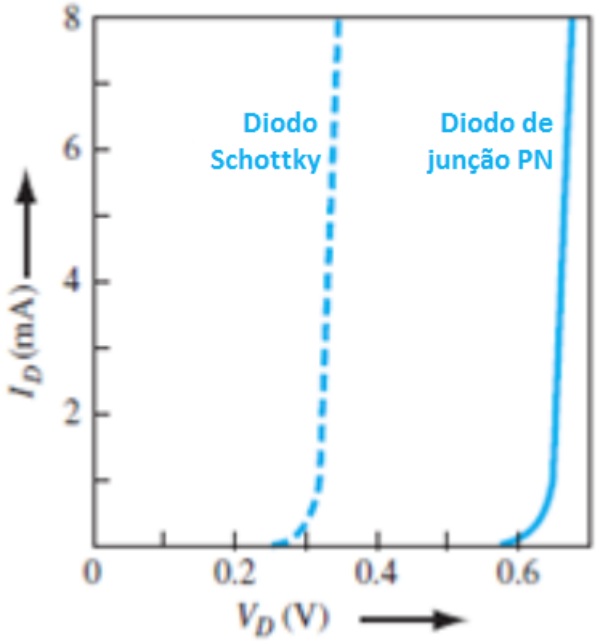
Cada um dos 400 diodos pode ser selecionado individualmente para se conectar com os pinos externos do circuito integrado (CI), que por sua vez serão ligados na estrutura de testes. Essa conexão interna é feita por meio de chaves construídas com transistores MOSFET, formando uma chave CMOS. Estes foram implementados com transistores de tipo I/O pois apresentam uma tensão de operação mais elevada (2.5V) e maior limite de tensão, resultando em menores vazamentos de corrente quando desligados. Vemos na Tabela 1 as dimensões usadas para os transistores das chaves, elas estão organizadas por geometria dos diodos, sendo o F na última linha referente a topologia *fingered*.

Tabela 1 – Dimensões dos transistores para as chaves CMOS.

	PMOS			NMOS	
Dimensões dos diodos	L	W	*fingers*	L	W
2μm x 2μm	0.24μm	0.84μm	1	0.24μm	0.84μm
5μm x 5μm	0.24μm	7.4μm	5	0.24μm	0.84μm
10μm x 10μm	0.24μm	22.8μm	10	0.24μm	0.84μm
10μm x 10μm - F	0.24μm	29.2μm	10	0.24μm	0.84μm

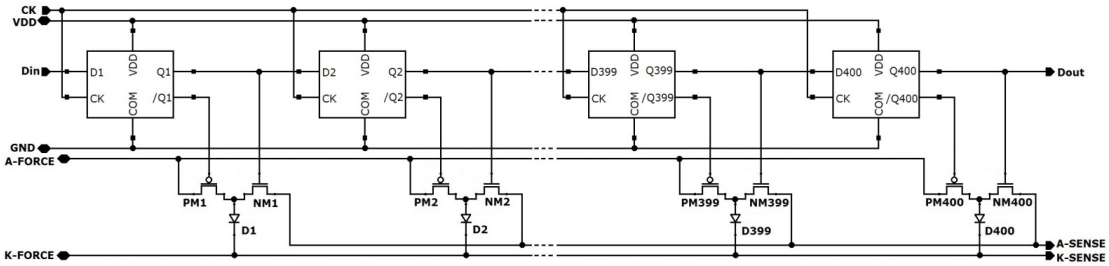
A necessidade de grupos com 100 diodos Schottky por chip surge do intuito de se objetivar obter os parâmetros de variabilidade do processo de fabricação com validade estatística. Com o elevado número de dispositivos de mesma geometria podemos considerar a incerteza comportamental como aleatória, nos habilitando a estimar a variabilidade por parâmetros como a média e desvio-padrão. Vemos na Figura 5 a curva dos diodo de junção PN, representada pela linha sólida, e de diodos Schottky, apresentada na forma da linha tracejada, é conhecida como curva I x V onde apresenta a tensão V no eixo X e a corrente I no eixo Y. O objetivo é levantar essas curvas aqui demonstradas para todas as geometrias com diferentes valores de temperatura, para posterior processamento. Nesse processamento busca-se obter os parâmetros de média e desvio-padrão para o par V x I dos diodos caracterizando a variabilidade de fabricação para cada variável estudada.

Figura 5 – Curva I x V dos diodos.



Como visto na Figura 6 há também o uso de flip-flops do tipo D associados com a estrutura de chaves CMOS. No circuito há um flip-flop do tipo D para cada um dos diodos, estes 400 flip-flops estão conectados em cascata serialmente para formar uma estrutura conhecida como *shift-register*. Cada um dos flip-flops comanda uma das chaves CMOS que estão conectadas com o lado do ânodo do diodo. Os flip-flops foram implementados utilizando transistores NMOS e PMOS com larguras de 0.84μm e 1.68μm respectivamente, mantendo-se o comprimento mínimo para o processo escolhido. Sabendo-se que a velocidade do clock seria baixa se implementou os flip-flops com *latches* estáticos.

Figura 6 – Chaves MOS e shift-registers conectados à matriz de diodos.



Os saídas Q_n dos flip-flops são responsáveis por ativar os transistores NMOS que conectam o ânodo do dispositivo sendo medido ao canal A-*sense* e as saídas $/Q_n$ ativam os transistores PMOS que conectam o ânodo do dispositivo sendo medido ao canal A-*force* como vemos na Figura 6. Ao carregarmos o *shif-register* com uma certa palavra válida de 400 bits, uma, e somente uma, chave CMOS, formada por par de transistores, um NMOS e um PMOS, está ativa. Essa chave CMOS conecta a rede de estimulo e medida do analisador de parâmetros ao dispositivo selecionado através das linhas A-*sense* e A-*force*.

Para a verificação da palavra de 400 bits usamos o pino de sinal D*out* conectado no fim do *shif-register*, por ele saíra a palavra do ciclo anterior para validação. Os pinos de sinal D*in*, D*out* and C *k* são bufferizados para evitar atrasos nos tempos de subida ou descida dos sinais. Deve-se notar também que o cátodo do diodo está conectado a outro par de canais *force-sense* também para evitar quedas de tensão.

2.4 Sistema de medição

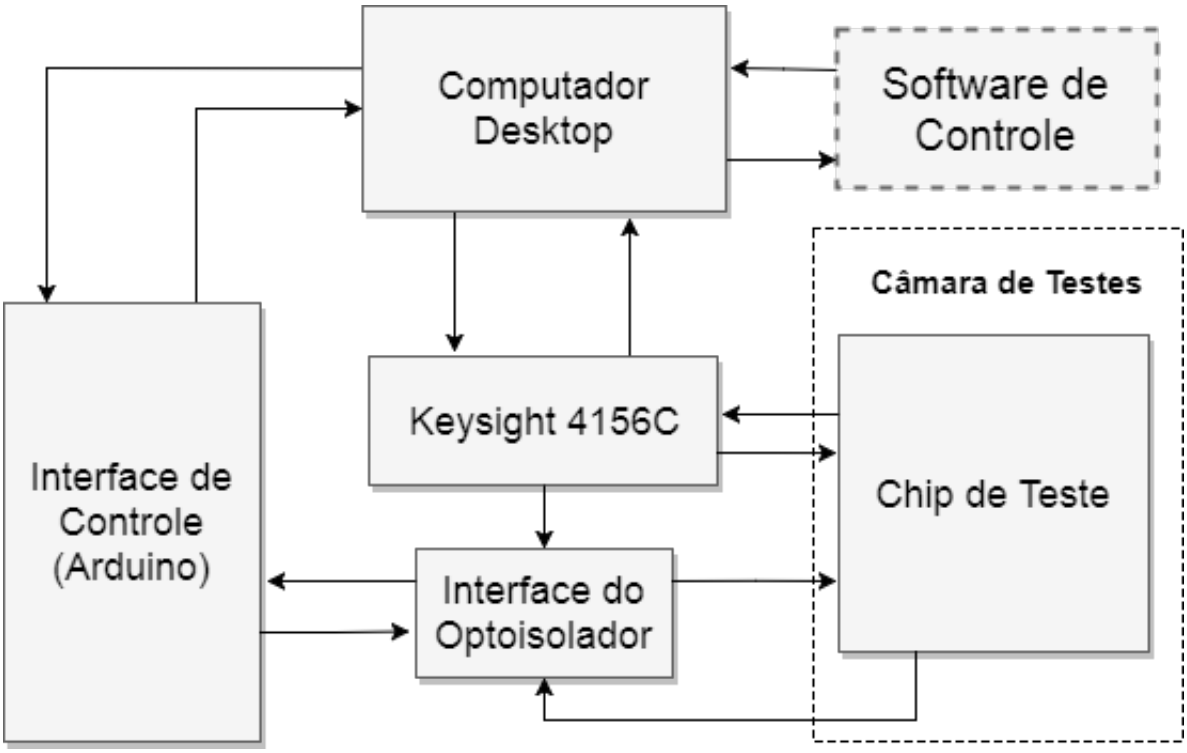
O sistema de medição desenvolvido pode ser visto na Figura 7, onde vemos as relações entre os diversos equipamentos e softwares que compõem o sistema e sua relação com o circuito integrado a ser medido. Vemos o chip de teste no interior da câmara de testes e que esta não se conecta com o restante da estrutura, sendo configurado manualmente.

O analisador de parâmetros (Keysight 4156C) se conecta de maneira bilateral com o chip de teste, essas conexões são responsáveis tanto pela alimentação do chip, vinda do analisador de parâmetros, assim como pelo envio dos estímulos e recebimento dos sinais medidos de tensão/corrente para a caracterização elétrica. Por sua vez a comunicação bilateral do mesmo com o computador desktop é feita pela interface GPIB/USB, onde o conector GPIB está do lado do analisador de parâmetros e a conexão no computador é feita por meio de uma porta USB. Essa conexão é responsável pela programação do analisador de parâmetros e recebimento dos dados medidos para armazenamento.

Há uma forma de comunicação entre o computador desktop e o software de controle, embora não física o software controla o que será enviado pelo hardware do computador por meio das suas conexões físicas e o que fazer com os dados recebidos. O computador desktop também se conecta com a interface de controle (Arduino) fazendo uso de uma conexão USB e é responsável pelo controle do dispositivo a ser medido, juntamente com a verificação da validade das medidas, contando com a conexão com o software de controle explicada anteriormente.

A interface optoisoladora possui comunicação bilateral com a interface de controle e com o chip de teste. Ela funciona como intermediária entre as duas partes para três sinais, esses sinais são os sinais de Clock, o sinal de dados que seleciona o dispositivo a ser medido e a verificação da palavra enviada ao *shift-register*. Desses sinais os dois primeiros se originam da interface de controle e o último é originado no chip de teste.

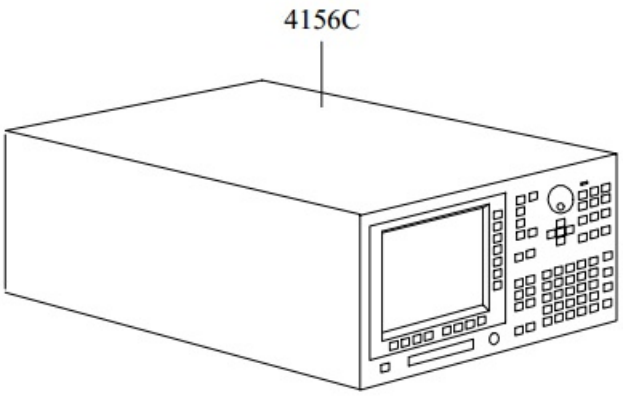
Figura 7 – Diagrama de blocos do sistema de medição.



2.4.1 Keysight 4156c

Para o sistema de medição e caracterização um dos equipamentos centrais é o analisador de parâmetros, sendo utilizado o Keysight 4156c, apresentado na Figura 8, que conforme (AGILENT TECHNOLOGIES, 2003) possui quatro unidades de estímulo e medida (SMUs) de alta resolução, duas unidades de fonte de tensão (VSUs) e duas unidades de monitoramento de tensão (VMUs) e é especialmente habilitado para medidas de baixa corrente e baixa tensão. O 4156c é capaz de medir valores de tensão com uma resolução de até 1 μV.

Figura 8 – Analisador de parâmetros 4156c



O 4156c é capaz de realizar três tipos de medições, sendo elas medição por varredura, medição por amostras e medida C-V quase estática. É possível também usar o botão rotativo do equipamento para realizar medições e ajustes rápidos visto em mais detalhes na Figura 9 que mostra uma vista frontal do equipamento.

Figura 9 – Painel do analisador de parâmetros 4156c

Para o armazenamento dos dados obtidos das medições, informações de configuração e informaões dos parâmetros do equipamento podem ser armazenados em um disquete de 3.5 polegadas ou em um dispositivo via LAN. A memória interna também pode ser usada para armazenamento, porém esse é limitado a quatro arquivos.

Outra qualidade é a capacidade de controle remoto do 4156c via GPIB, permitindo três tipos de conjunto de comandos, sendo 4155C/4156C SCPI (*Standard Commands for Programmable Instruments*), 4155C/4156C FLEX (*Fast Language for Execution*) e a sintaxe de comando 4145B. Há também a possibilidade de uso do 4156c de maneira independente para o desenvolvimento e execução de programas, visto que o mesmo possui *Instrument* BASIC.

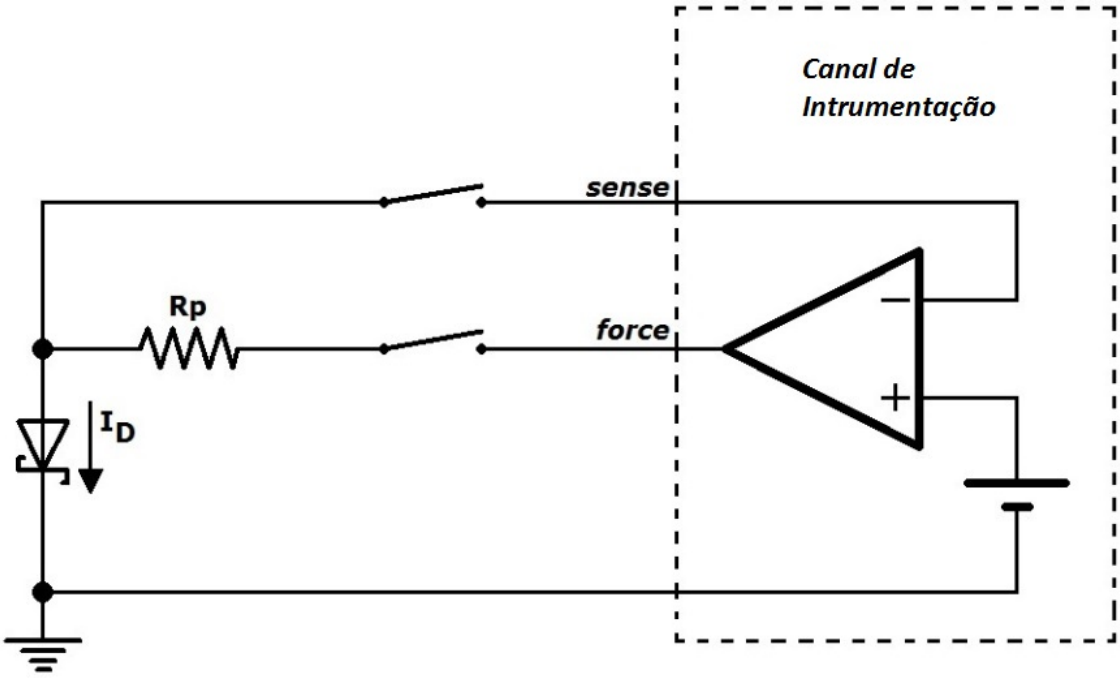
2.4.1.1 Force-sense

O circuito a ser medido requer que o analisador de parâmetros possua a função *force-sense*, cujas conexões se assemelham as da medida a quatro fios. O sistema *force-sense* coloca uma certa tensão (ou corrente) conhecida pelo *force* e estima a corrente (ou tensão) resultante pelo conector *sense*, que funciona como um sinal de realimentação. É importante que o conector *sense* esteja posicionado o mais próximo possível do ponto de medida.

Essa técnica é importante pois reduz os erros de medida principalmente para medições de altas correntes ou baixas resistências. O caminho de medida entre o analisador de parâmetros e o dispositivo a ser medido tem uma resistência residual, essa resistência torna-se não desprezível quando o valor da mesma se aproxima do valor a ser medido, o que ocasiona erros de medição. Se o *sense* for conectado com o *force* o mais próximo possível do dispositivo medido, o circuito irá funcionar estabilizando o ponto de medida na tensão configurada, fazendo com que a queda de tensão seja desprezível.

Conforme vemos na Figura 10, ambos os canais, *force* e *sense*, irão formar o canal analógico, o primeiro sendo responsável por excitar o nó que será testado e o segundo sendo responsável por medir a tensão efetiva nos terminais do dispositivo. A resistência Rp demonstra o efeito combinado das chaves MOS que estão implementadas no circuito a ser testado. Essa abordagem também é conhecida como estratégia de medida Kelvin.

Figura 10 – Técnica de medição conhecida como force-sense.

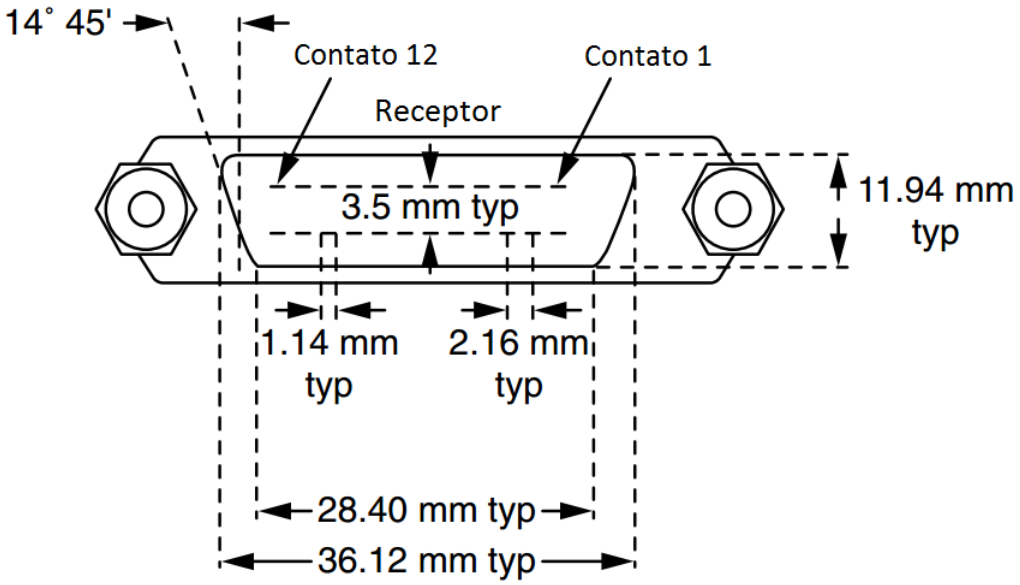


2.4.1.2 GPIB *General Purpose Interface Bus*

O padrão para barramento de comunicações digitais IEEE-488, também geralmente conhecido como HP-IB (*Hewlett-Packard Instrument Bus*) e GPIB (*General Purpose Interface Bus*), é um barramento paralelo de 8 bits. O GPIB permite que até 15 equipamentos sejam ligados simultaneamente no mesmo barramento paralelo.

Em 2004 o IEEE e a IEC combinaram os seus padrões de modo a ter um padrão única IEEE/IEC devido as diversas versões vindas desde 1975. Esse padrão unificado é definido como IEC-60488-1 que define os aspectos mecânicos, elétricos, e de protocolo do GPIB e o padrão EIC-60488-2, complementar a esse, que define os convenções de sintaxe, formato e comandos. Sendo dois padrões distintos, é possível seguir um sem seguir ao outro, particularmente seguir o IEC-60488-1 sem seguir o IEC-60488-2.

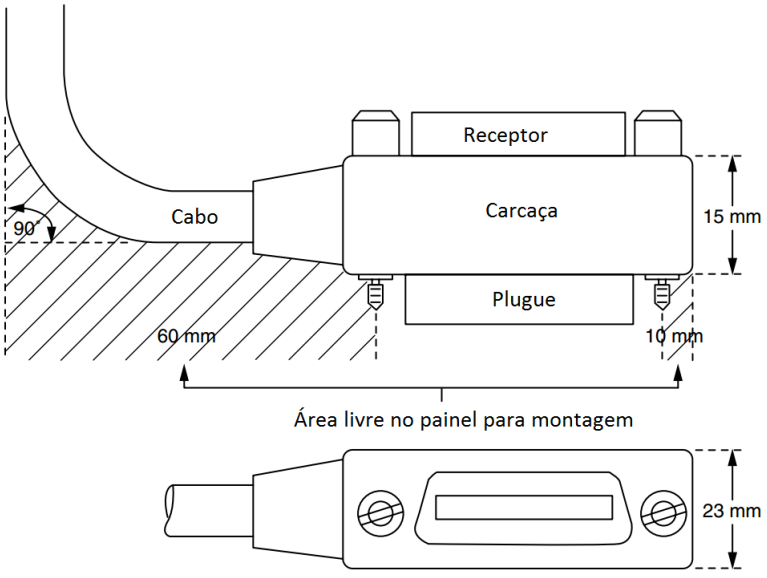
Figura 11 – Conector GPIB fêmea.



Fonte: (IEEE, 2004)

O conector GPIB possui 16 linhas de sinal, sendo 8 usadas para transferência bidirecional de dados, 3 para *handshake*, que, basicamente, é o processo onde duas máquinas informam uma a outra que se reconheceram e estão prontas para iniciar comunicação, e 5 para gerenciar o barramento, além de 8 linhas de sinal para terra (GND). Podemos ver na Figura 11 o formato em D do conector com as medidas típicas. Tipicamente o conector GPIB é apresentado com conectores macho e fêmea na mesmo estrutura como apresentado na Figura 12

Figura 12 – Conector GPIB.



Fonte: (IEEE, 2004)

2.4.2 Câmara Térmica

O objetivo do sistema de testes é caracterizar a variabilidade dos diodos Schottky em função da sua geometria (diferentes tamanhos de dispositivo), da sua polarização (pares V x I) e da temperatura. Esses resultados devem ser confiáveis e estatisticamente válido, para tal é imperativo que o ambiente onde os testes são realizados seja controlado. Para se obter um ambiente controlado utilizamos uma câmara de testes, sendo escolhida para essa estrutura a câmara térmica Tenney TU-Jr da *Thermal Product Solutions* (TPS).

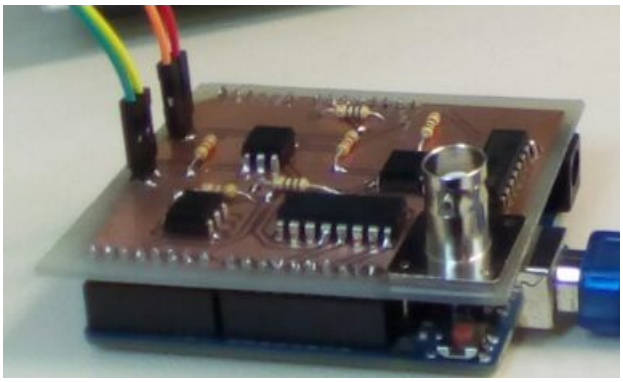
Segundo (THERMAL PRODUCT SOLUTIONS,) o equipamento escolhido conta com uma tolerância no controle de $\pm 0.3^oC$ após a estabilização, alcançando isso por meio do uso de um sensor RTD (*Resistance Temperature Detector*) de platina e uma resolução de $\pm 0.1^oC$ para ambos, o mostrador e o valor-alvo. A temperatura no interior da câmara pode ser configurada para alcançar valores entre -75^oC e 200^oC. Para que se mantenham condições uniformes no interior da câmara a mesma faz uso de condições de recirculação com uma corrente vertical de ar de cima para baixo. Na Figura 13 podemos ver na porta a janela para visualização do dispositivo sendo medido.

Figura 13 – Câmara térmica Tenney TU-Jr.

2.4.3 Arduino

Como dispositivo para realizar a interface entre o computador e o chip a ser medido foi utilizado uma placa Arduino Uno pela sua facilidade de aquisição e compatibilidade. A estrutura Arduino Uno é conhecida como uma placa de prototipação e desenvolvimento, sendo adequada para o uso do desenvolvimento dessa estrutura. O seu uso como interface leva os dados de clock e de qual dispositivo será medido para o chip, além de receber o sinal para verificação e validação dos dados medidos. Vemos na Figura 14 a placa do Arduino Uno com a estrutura da interface optoisoladora.

Figura 14 – Arduino Uno com interface optoisoladora



Ao fim da etapa de desenvolvimento a estrutura do Arduino permite que seja confeccionada uma placa mais simples onde será colocado seu microprocessador, inclusive integrando tal placa com a estrutura optoisoladora. A robustez e versatilidade da sua estrutura faz o Arduino uma escolha natural para o uso como interface nesta estrutura de testes.

3 SOFTWARES DESENVOLVIDOS

Para que a plataforma de testes seja implementada há a necessidade de desenvolvimento de alguns softwares responsáveis pela configuracão e controle não somente do analisador de parâmetros, como também da estrutura a ser testada. Foi desenvolvido um software em C# para, através de uma interface Arduino, realizar o controle do circuito integrado a ser testado, como a seleção do dispositivo a ser medido. O desenvolvimento da configuração do analisador de parâmetros seguiu diretrizes mais delimitadas pela documentação do mesmo.

3.1 Interface de Controle

O dispositivo a ser medido é selecionado por meio de chaves CMOS e um *shift-register*, este será carregado com uma palavra de 400 bits. A geração dessa palavra é feita pelo computador através de um software criado em C# cuja interface gráfica é vista na Figura 15.

Figura 15 – Interface gráfica entre o Arduino e o usuário.

O programa cria uma interface entre o usuário e o programa gravado no Arduino através

de uma comunicação serial com o computador. Essa permite ao usuário selecionar que plataforma de Arduino será conectada ao programa, qual é o próximo dispositivo a ser medido e também realizar uma verificação do sinal do vetor enviado. Essa verificação é para garantir que a palavra enviada confere com a palavra que foi recebida pelo circuito a ser medido. É importante resaltar que tal verificação só é possível devido a estrutura do *shift-register* projetado e descrita em 2.3.

O processo de verificação consiste em enviarmos serialmente duas vezes consecutivas a mesma palavra contendo a informação do dispositivo selecionado. Após enviarmos a primeira palavra, essa sera armazenada no *shift-register* e quando essa for deslocada para frente, continuando o processo, teremos exatamente a mesma palavra na saída do *shift-register*. Esse sinal será tratado utilizando-se uma interrupção do Arduino, comparado com a palavra que deveria ser enviada ao sistema e retorna o status da operação ao usuário, indicando se algo não aconteceu da maneira esperada. Este processo, embora simples, é de extrema importância para a qualidade dos dados gerados das medições.

O sinal sempre será composto por apenas um bit em nível alto, que em cada medição diferente será deslocado até a posição correta correspondente. Além do sinal a ser deslocado o código também é responsável por gerar um sinal de *clock* destinado a deslocar corretamente o sinal com a informação. Dentro do programa ainda podemos definir a frequência do sinal de *clock* além do tipo de sinal a ser enviado com a informação. Para o presente projeto é utilizado um contador 2^n, o que permite somente um sinal em nível alto, porém é possível também definir diferentes contadores binários para uso em futuros projetos, como contadores sequenciais para conversores analógico/digital e digital/analógico.

3.1.1 Programa de Controle

A interface entre o Arduino e o computador foi escolhida devido ao fato de ser mais amigável ao usuário para selecionar os parâmetros de medida e exibir possíveis alertas e erros do sistema. A aplicação desenvolvida para Windows usa a plataforma .NET com o auxílio do Visual Studio da Microsoft na linguagem C#. O programa é composto por um terminal simples que envia e recebe informações através de uma interface gráfica apresentada na Figura 15.

A comunicação entre a placa Arduino e o computador é realizada através da porta serial emulada através do driver da USB. No botão conectar é inserido um componente timer que é responsável pela atualização das portas COM disponíveis no computador, isso ocorre a cada 1s. O próprio ambiente Visual Studio tem um componente de comunicação serial que pode ser usado para a configuração dessa rotina. Antes de conectar a serial é necessário verificar as portas COMs disponíveis e em qual porta o usuário deseja realizar a conexão. Para isso há o método privado dentro da classe principal, chamado de atualizaListaCOMs, que verifica se o numero de portas disponíveis mudou e caso tenha mudado atualiza uma lista com as portas disponíveis. Lembramos que só há uma COM disponível caso uma placa Arduino esteja conectada. Para que esta atualização seja automática é usado o timerCOM habilitado anteriormente para gerar este

evento a cada 1s.

O evento associado ao clique do botão "Conectar"será usado para estabelecer a conexão e também servirá para desconectar quando uma conexão estiver ativa. Foi tomado o cuidado de inserir uma proteção para que o programa não feche deixando a porta COM aberta, isso impediria a sua utilização por outros programas. A utilização do C# contribui para a facilidade de implementação, bastando criar um evento associado ao fechamento da janela perguntando se a porta serial está aberta e, caso seja positivo, o comando serialPort1.Close(); é executado ao fecharmos o programa. Agora os dados recebidos pela serial são guardados em uma variável global do tipo *string*.

Através do botão selecionar, a ação do clique está configurada para que seja enviado o valor inserido no campo adjacente, neste campo tomamos o cuidado de limitar o tipo e a faixa de valores que podem ser inserido pelo usuário para que sistema de validação não encontre problemas. Prevendo futuras atualizações e maior gama de funcionalidades, junto com o valor do dispositivo selecionado é enviado um caractere que conecta essa mensagem enviada ao Arduino à ação de gerar o vetor e enviar os dados para a saída digital. Essa configuração possibilita que no futuro outros botões sejam responsáveis por ações diferentes no microcontrolador, como gerar outro tipo de vetor de sinais de teste.

No outro lado o Arduino esta configurado para receber a informação, decodifica-la e realizar as ação requeridas. Neste programa, ele divide a palavra recebida em duas partes, a primeira que identifica o bloco de ações requisitadas e a segunda traz um complemento a estas ações. Para o caso atual, o complemento se refere à posição do dispositivo que será selecionado e o bloco de ações constitui:

1. Geração do sinal de teste;
2. Envio do sinal de teste saída digital (enviando duas vezes para validação);
3. Recebimento do sinal de validação (interrupção);
4. Envio da informação necessária para a porta serial;

Concomitante com o último passo o programa em C# confere com o sinal requerido pelo usuário, gerando um aviso de confirmação ou um alerta indicando possível erro ou desconformidade.

3.2 Programa de Template

De forma a deixar o sistema de testes mais modular e com maior probabilidade de reuso em futuros projetos foi implementado um programa de template antes do programa de medição.

Tal programa é apresentado no Apêndice B e foi implementado em C++ e busca sequencialmente executar as determinadas tarefas:

1. Declarar as funões e definir os macros do analisador de parâmetros;
2. Checar por erros do instrumento e dispositivos;
3. Declarar o subprograma que vai ser definido no programa de medição;
4. Estabelecer a conexão com o analisador de parâmetros (incluindo tipo de conexão e endereço GPIB);
5. Reiniciar o analisador de parâmetros após a conexão;
6. Determinar o *timeout* dos *drivers* de entradas e saídas;
7. Habilitar a verificação automática de erros;
8. Testar a verificação automática de erros;
9. Após os testes, encerrar a conexão com o analisador de parâmetros;
10. Verificar se houver erros ao encerrar a conexão.

Este programa serve então como base, ou template, para a construção do programa de medição descrito em 3.3 de modo que é necessário somente alterar o subprograma do mesmo. O programa de template tem por propósito também testar o ambiente de compilação e execução do programa de medição, visto que as configurações de caminhos e requisitos de ambos são idênticas.

Segundo (AGILENT TECHNOLOGIES, 2004) para que possam ser executados os programas referidos ao analisador de parâmetros é necessário que sejam adicionados:

1. No caminho de procura de arquivo:
   - diretório que inclui o arquivo "hp4156b.h"e os arquivos relativos ao *include* do VISA (ex.: </ProgramFiles/VISA/winnt/include>).
2. No caminho para procura de bibliotecas:
   - diretório que armazena o arquivo hp4156b.lib e or arquivos de biblioteca relativos ao VISA (ex.: </ProgramFiles/VISA/winnt/lib/mscforMicrosoftVisualC++> ou </ProgramFiles/VISA/winnt/lib/bcforBorlandC++Builder>)
3. Bibliotecas adicionais para o *Linker* do projeto
   - hp4156b.lib

De maneira a deixar a estrutura de medições mais robusta, compacta e modular todos os arquivos citados como necessários foram posicionados dentro de um mesmo diretório, facilitando o compartilhamento e futuros trabalhos no mesmo. Após o programa de medições montado e compilado não há a necessidade da configuração dessas etapas para a execução e uso da estrutura de caracterização.

3.3 Programa de Medição

O programa de medição tem como requisito o programa de template descrito em 3.2, isso permite alterar o protocolo seguido para as medições pelo analisador de parâmetros mais rapidamente. Conforme (AGILENT TECHNOLOGIES, 2004) para a concepção do programa de medição usando como base um programa de *template* devemos usar os seguintes passos:

1. Planejar as medições automáticas para decisão dos seguintes itens:
   - Dispositivo medido: discreto, encapsulado, no wafer, etc.
   - Parâmetros ou características a serem medidas: tensão de *threshold*, fator de multiplicação de corrente, etc.
   - Método de medição: medição pontual, varredura linear, etc.
2. Copiar e renomear o arquivo do programa de *template* (o projeto, não o arquivo individual para não perder as configurações prévias dos caminhos);
3. Abrir o projeto;
4. Abrir o arquivo fonte do item 3.2 e preencher a sub-rotina *perform_meas*. Então executar as funções do driver do analisador de parâmetros para definir as entradas e saídas do mesmo;
5. Inserir o código para exibir, armazenar ou processar os dados dentro da sub-rotina;

Para a sub-rotina *perform_meas* foi implementado o protocolo relativo a varredura de tensão e medição das correntes correspondentes, obtendo assim um par tensão-corrente. Na sub-rotina também se define o diretório onde serão salvos os dados medidos.

4 HARDWARE

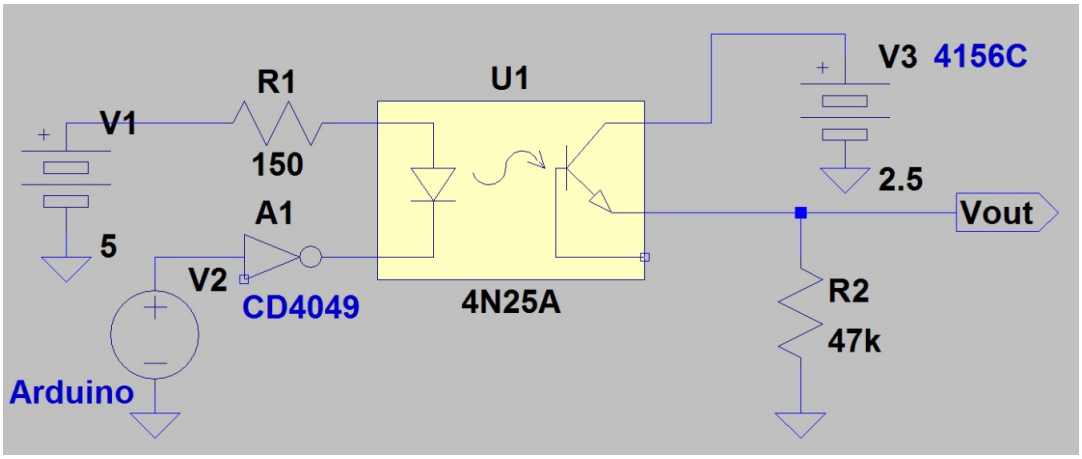
Além do desenvolvimento de softwares necessários para controle do sistema, medição e armazenagem dos dados é necessário o desenvolvimento de alguns hardwares para garantir a segurança dos equipamentos e circuitos. Uma prática necessária ao se lidar com circuito sensíveis é a isolação galvânica do mesmo. Posto isso, foi desenvolvida uma interface optoisoladora para proteger o circuito de sobretensões e possíveis interferências vindas do computador no processo de medição, como visto em (KLIMACH, 2008).

4.1 Interface Optoisoladora

A interface optoisoladora tem por objetivo proteger o circuito contra sobretensões e interferências por meio de isolação galvânica. Além de prover a isolação do circuito de teste ela também é responsável por baixar o nível de tensão que chegará ao circuito a ser testado. A interface de controle conectada por meio de um Arduino Uno tem seu nível alto em 5V, um valor que pode facilmente danificar os vários dos circuitos desenvolvidos pelo grupo e presentes no mesmo chip fabricado.

Podemos ver na Figura 16 que o circuito é simples, sendo composto pelo optoisolador, um inversor lógico e resistores para limitação das correntes. A topologia escolhida é feita para que não haja inversão da polaridade do sinal entre os lados da isolação. Foi utilizado o circuito integrado 4N25 como optoisolador, os resistores R1 e R2 garantem a sua polarização. O circuito integrado 4N25 foi escolhido pela facilidade de acesso à ele, experiência prévia com o mesmo e as suas características, como a diversidade de aplicaçõces.

Figura 16 – Diagrama elétrico individual da interface optoisoladora.



O inversor usado é o CD4049, este é um inversor lógico que possui seis entradas e saídas individuais, a sua escolha passa, além da possibilidade de acesso à ele, pela necessidade do

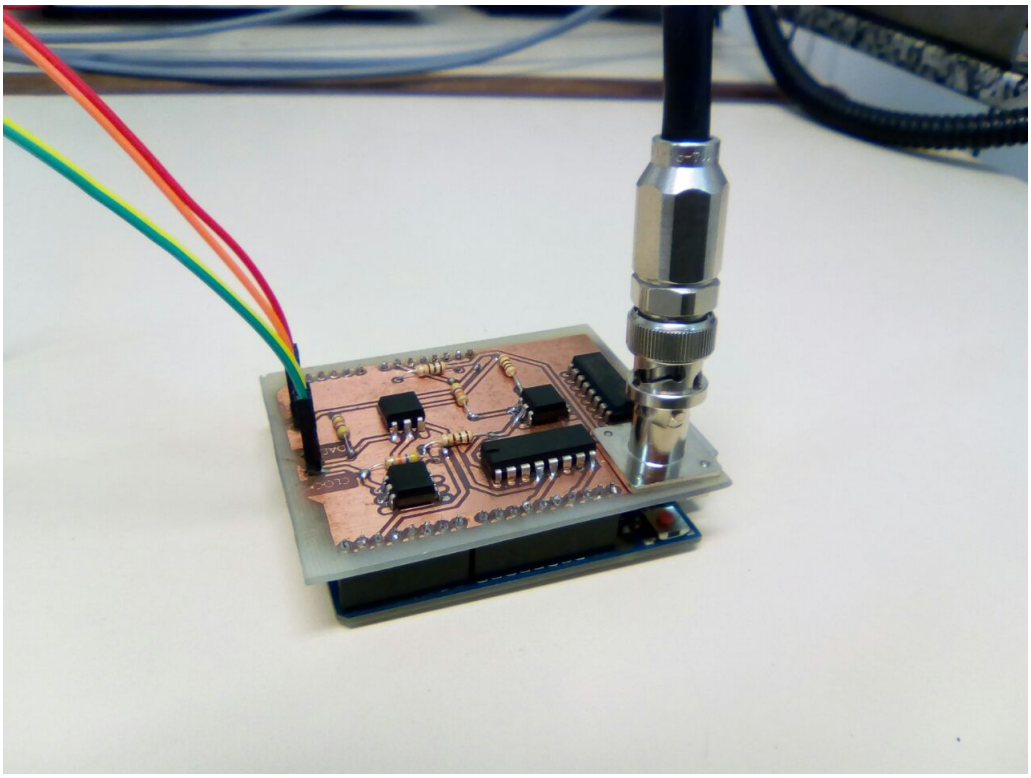
circuito inversor ser da tecnologia CMOS. O circuito apresentado na Figura 16 foi replicado três vezes, visto que há a necessidade de isolar três sinais, sendo eles o Clock, o sinal de dados que seleciona o dispositivo a ser medido e a verificação da palavra enviada ao *shift-register*. Este último é enviado do circuito integrado medido para o Arduino, sendo necessário um aumento de tensão do mesmo ao invés de uma atenuação.

O nível de tensão na saída do circuito optoisolador é alcançado ao conectarmos o coletor e o emissor do seu fototransistor interno aos dois canais do analisador de parâmetros, estes dois canais são responsáveis por fornecer o nível de tensão desejado para que o circuito sendo medido possa operar. Para alcançarmos satisfatoriamente os níveis de tensão em ambas extremidades do circuito optoisolador será necessário que utilizemos dois circuitos de terra isolados entre si.

4.1.1 Confecção

Após o projeto satisfatório da interface optoisoladora, a confecção da mesma seguiu algumas diretrizes importantes para a melhor performance experimental. Buscando reduzir as interferências e ruídos provenientes dos cabos e conexões entre a placa da interface de controle (Arduino) e a placa de circuito impresso contendo os optoisoladores, essa foi construída de forma a encaixar perfeitamente sobre a placa do Arduino.

Figura 17 – Interface optoisoladora confeccionada.



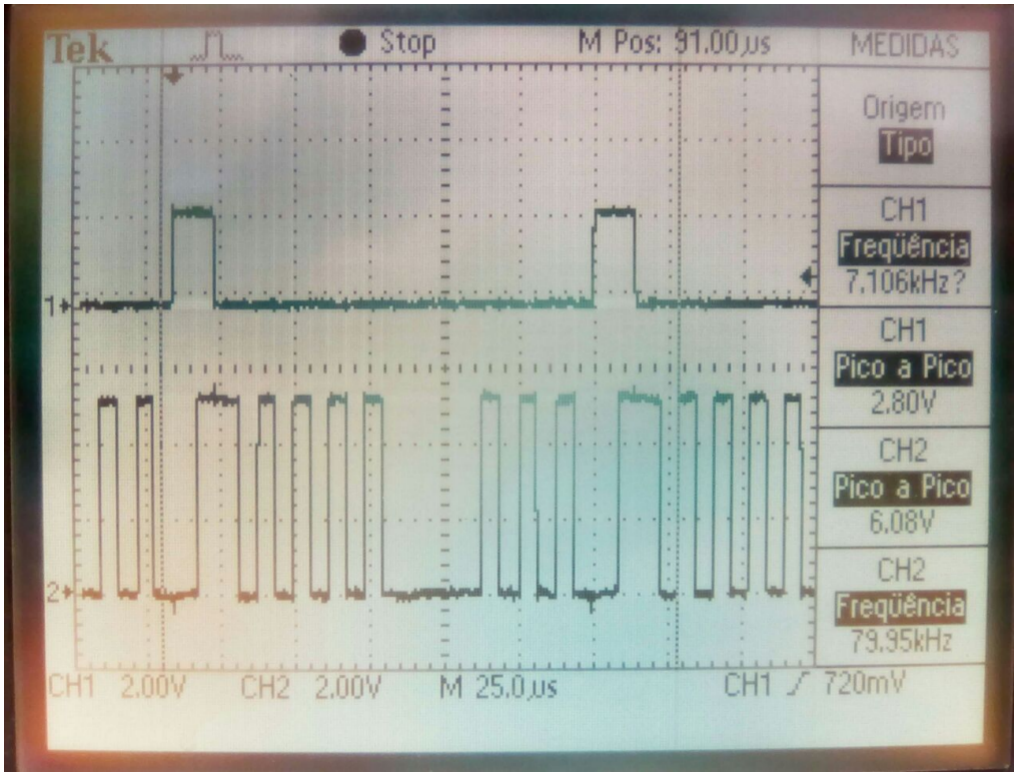
A necessidade do uso de dois circuitos de terra isolados ocasionou a escolha de uma placa com duas camadas de cobre, apresentada na Figura 17. Após o posicionamento e solda

dos componentes na mesma foi adicionada uma camada de verniz para proteção e isolação da superfície do cobre. A confecção da mesma foi posteriormente validada.

4.1.2 Testes Preliminares

Os testes preliminares da interface optoisoladora foram realizados com um vetor de 8 bits ao invés de 400, que será usado na versão final da estrutura, porém essa característica é facilmente escalável. Vemos na Figura 18 dois sinais, na parte inferior vemos o sinal de clock que vem da interface de controle indo para o circuito a ser testado. O sinal foi medido antes da interface optoisoladora para que a sua amplitude pudesse ser comparada com o sinal na parte superior.

Figura 18 – Testes preliminares da interface optoisoladora confeccionada.



Na parte superior é apresentado o sinal de informação, este que é responsável pela seleção do dispositivo a ser testado e/ou medido. Vemos que a amplitude do sinal de informação é significativamente menor em comparação com o sinal de clock. A razão para tal diferença de amplitudes é a segunda função da interface optoisoladora, a de regular e controlar o nível de tensão do sinal de saída. O circuito optoisolador utilizado possui controle independente do nível de tensão do sinal de saída. A referência para essa tensão de saída será fornecida pelo analisador de parâmetros visando prevenir sobretensões nos sinais que chegam ao circuito sendo testado.

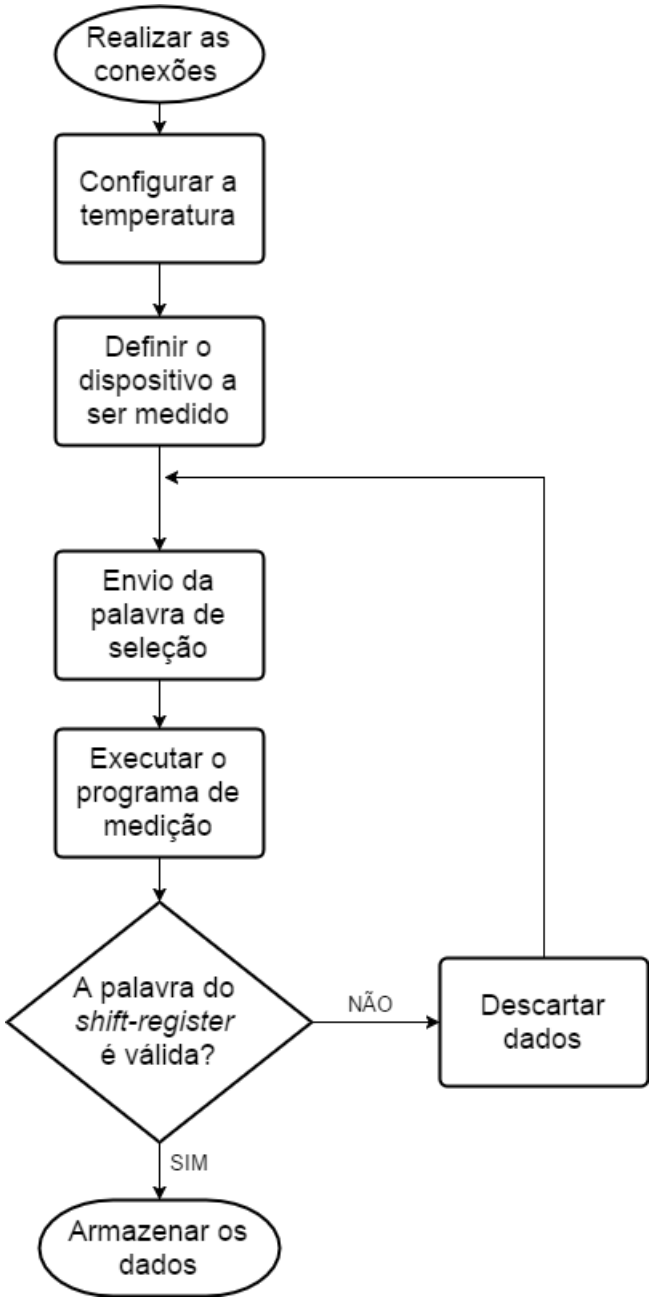
4.2 Computador

Há a necessidade de uso de um computador para funcionar como interface de controle entre o usuário e o Arduino, sendo o meio por onde o usuário é alertado de eventuais erros e falhas que podem ocorrer durante o envio do sinal para o teste. Além disso o analisador de parâmetros é conectado ao computador por meio da interface GPIB do 4156C e USB no computador, fazendo-se o uso de um adaptador. A conexão entre o computador e o analisador de parâmetros nos permite armazenar os dados medidos diretamente em arquivos de texto para posterior processamento e administração de todos conforme (K.TO, 1974).

5 PROTOCOLO DE MEDIÇÃO

As medições seguem o protocolo exposto na Figura 19 onde após realizadas as conexões necessárias é configurada manualmente a temperatura da câmara de testes descrita em 2.4.2. Nas medições para os diodos Schottky serão usadas temperaturas entre -20^oC e 100^oC. Está implícita nessa etapa a necessidade de permitir que a câmara alcance e estabilize a temperatura interna para a temperatura configurada, esse tempo pode variar devido a fatores externos.

Figura 19 – Fluxograma do protocolo de medição.



Após a temperatura definida, a próxima etapa consiste em determinar o dispositivo a ser medido, essa configuração é feita na interface apresentada em 3.1 onde se define o número correspondente ao diodo Schottky que será caracterizado. Em seguida ocorre o envio da palavra de seleção para o *shift-register*, esse envio passa pela interface GPIB descrita em 2.4.1.2. É executado o programa de medição apresentado em 3.3, seguido pela verificação da palavra no *shift-register*, se a mesma for válida, os dados são armazenados para processamento posterior. Caso a palavra não seja válida, os dados são descartados e o protocolo retorna para a etapa de envio da palavra de seleção.

6 Conclusões

O objetivo do trabalho aqui descrito consiste primeiramente em estudar o equipamento a ser utilizado, seguido da implementação e desenvolvimento da estrutura de testes e por fim realizar medidas para validação da estrutura. Os objetivos não foram plenamente alcançados pois não foi possível a integração de todas as partes individuais com um teste de caracterização de um dispositivo devido a restrições de tempo e disponibilidade dos equipamentos de teste. O estudo dos equipamentos teve resultado satisfatório onde foram possíveis sanar diversas dificuldades quanto a compatibilidade dos equipamentos.

A implementação galgou resultados favoráveis tanto em hardware quanto em software. No âmbito do hardware, a interface optoisoladora se destaca pela sua modularidade e pela contribuição para futuros projetos, sendo uma estrutura já testada e validada. No âmbito do software vemos que a interface de controle, também testada e validada, funciona de maneira robusta e pode ser facilmente integrada com programas de medição. Tal programa de medição pode ser o desenvolvido preliminarmente aqui ou algum futuro programa de medição desenvolvido com base no programa de template, implementado, descrito e validado aqui.

Com respeito dos requisitos para a integração, se estabeleceu aqui quais as diretrizes e arquivos necessários para tal integração, tanto os individuais quanto a compatibilidade entre as partes. Foi organizado também um diretório com os arquivos necessários de forma a facilitar e habilitar futuros usos, visto que a obtenção de alguns desses arquivos necessitava de softwares já não mais usuais no mercado e com os quais não se tem mais compatibilidade.

As estruturas descritas e desenvolvidas no decorrer desse trabalho, embora ainda não integradas, foram projetadas para habilitar tal integração. Como mostrado nesse trabalho e em (PEDRINI THALES STEDILE, 2016), a previsão de certas funcionalidades na etapa de projeto são extremamente importantes para que o projeto possa ser desenvolvido.

Salienta-se, também, que este trabalho produziu o artigo intitulado "An Automatic Test System for Schottky Diode Variability Characterization"(em inglês: Um Sistema de Testes Automático para Caracterização de Variabilidade de Diodos Schottky) aceito para ser apresentado no SFORUM 201 - 17^{th} Microelectronics Students Forum (em inglês: 17^{o} Fórum de Microeletrônica para Estudantes) que acontece durante o Chip on the Sands 2017. Este é um evento internacional, sendo um dos maiores, ou o maior, simpósio de micro e nano eletrônica da América Latina, o que fornece credibilidade aos resultados apresentados por esse trabalho.

7 Propostas de Trabalhos Futuros

O trabalho aqui descrito habilita diversos trabalhos futuros, as estruturas aqui desenvolvidas e implementadas possuem grande possibilidade de reuso, inclusive na continuação direta das pesquisas aqui citadas, entre eles:

- Integração das diversas estruturas aqui desenvolvidas e caracterização da matriz de diodos Schottky;
- Uso da interface de controle para testes de dispositivos D/As;
- Validação de lotes de dispositivos adquiridos.

Referências

AGILENT TECHNOLOGIES. *User's Guide Volume 1 General Information*. [S.l.], 2003. Citado na página 23.

AGILENT TECHNOLOGIES. *VXIplug&play Driver User's Guide*. [S.l.], 2004. Citado 2 vezes nas páginas 32 e 33.

IEEE. *Higher performance protocol for the standard digital interface for programmable instrumentation - Part 1: General*. [S.l.], 2004. Citado na página 26.

KLIMACH, H. *Modelo de Descasamento (Mismatch) Entre Transistores MOS*. Tese (Doutorado) — Universidade Federal de Santa Catarina, 2008. Citado 4 vezes nas páginas 13, 14, 19 e 34.

K.TO, R. E. T. Instrumentation: Automatic test systems: Seven basic elements make up the ideal ats; neglect of any one is an invitation to disaster. v. 11, n. 11, p. 44–52, 1974. Citado na página 37.

PEDRINI THALES STEDILE, Y. C. W. M. Test matrix for schottky diode variability characterization. *SFORUM 2016*, 2016. Citado 5 vezes nas páginas 14, 15, 18, 19 e 40.

SCHMID, A. H. H. Measuring a small number of samples and the 3v fallacy: Shedding light on confidence and error intervals. *IEEE Solid-State Circuits Magazine*, v. 11, n. 6, p. 52–58, 2014. Citado na página 13.

SZE, S. M.; NG, K. K. *Physics of Semiconductor Devices*. 3ª edição. ed. [S.l.]: Wiley-Interscience, 2006. Citado 2 vezes nas páginas 13 e 14.

THERMAL PRODUCT SOLUTIONS. *Tenney Jr.Test Chambers*. [S.l.]. Citado na página 27.

TIETZE, U.; SCHENK, C.; GAMM, E. *Electronic Circuits: Handbook for Design and Application*. 2ª edição. ed. [S.l.]: Springer, 2008. Citado 3 vezes nas páginas 16, 17 e 18.

V. HAMILTON KLIMACH, S. B. R. C. Nano-watt 0.3 v supply resistorless voltage reference with schottky diode. *LASCAS 2016*, v. 1, p. 175–178, 2016. Citado 2 vezes nas páginas 13 e 14.

APÊNDICE A – Código Interface Controle

Programas/arduino_officialV2.ino

```
1  // teste para gerar o sinal de acionamento para matriz a partir
       de um valor lido na serial
2  // com o monitor serial para verfificar o que anda sendo lido
       pelo programa.
3  // xxxxxxxxxxxxxxxxxxx -- DEBUG TEST --
       xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
4  // ok, envia msg pelo form e liga o LED correspondente,
       interface do form refeita, limitacoes
5  // nos valores de selecao do dispositivo ok!!
6  // a fazer: valida\c{c}\~{a}o do sinal enviado atraves da
       interrup\c{c}ao INT0 - disponivel no pino D2 do
7  // arduino UNO. Ligaremos ao pino D2 do arduino a saida do SR,
       contando o numero de pulsos
8  // no pino CLK do SR, acionaremos a interrupção externa por
       borda de subida, interrompendo esta
9  // contagem, este valor indicará a quantidade de posiçoes
       deslocadas pelo SR, que somando +1
10 // nos informa a posicao do bit acionado no SR.
11
12 #include <string.h>
13
14 #define valida 2 //interrupção
15 #define atualiza  12 //Atualiza saida do SR 74HC595
16 #define data   11 //Entrada de dados
17 #define clk  10 //Registrador de deslocamento CLK
18 #define freqClk 10000 // 10kHz
19 #define tamREG 8 // tamanho da palavra, para os teste SR tem
       8bits..
20
21 String comando;
22 String complemento;
23 String info;
24 float tONoff = (1/freqClk)*(1000000/2); //meio periodo em
       microsegundos
25 int diodoSelecionado; //variavel recebida do PC, que informa o
       dispositivo a ser medido
```

```
26 unsigned int verifCont=0;       //contador de pulsos do sinal de
      CLK
27 unsigned int dispValidado=0;    // dispositivo efetivamente
      acionado pelo sistema
28
29 void enviaSinal(int diodoSelecionado);  // Protótipo da função
      do registrador
30 void criaPeriodo(int porta);            // função gera 1 periodo
31 void interrupt()
32 {
33   dispValidado=verifCont;
34 }
35
36
37 void setup()
38 {
39   Serial.begin(9600);  //inicia comunicação serial com 9600
40   pinMode(valida, INPUT);
41   pinMode(atualiza, OUTPUT);
42   pinMode(data,  OUTPUT);
43   pinMode(clk, OUTPUT);
44   attachInterrupt(0, interrupt,RISING); // interrupção INT0
        borda de subida
45   enviaSinal(0);
46   verifCont=0;
47   dispValidado=0;
48 }
49 //---------------------------------------------------------------------
50 // PROGRAMA PRINCIPAL
51 //---------------------------------------------------------------------
52 void loop()
53 {
54   dispValidado=0;
55   verifCont=0;
56   if(Serial.available())         //se algum dado disponível
57    {
58     info = Serial.readString();   //le String enviada pelo PC
59     comando = info[0];
60     complemento = info.substring(2, 5);
61     diodoSelecionado=complemento.toInt();
62     //Serial.print(comando);           //retorna o que foi lido
63     }
```

```
64
65    if (comando == "D")
66    {
67     enviaSinal(diodoSelecionado);
68     enviaSinal(diodoSelecionado);
69     dispValidado=(dispValidado+3)/2;      //ajuste ao valor
          selecionado
70     Serial.print(dispValidado);           //retorna o que foi
          lido
71     //verifCont=0;
72     comando="0";
73     dispValidado=0;
74    }
75 }
76
77
78
79 //-------------------------------------------------------------------------
80 // FUNÇÕES
81 //-------------------------------------------------------------------------
82 void criaPeriodo(int porta) // duty cicle de 50%
83 {
84   digitalWrite(porta,HIGH);
85   delayMicroseconds(tONoff);
86   digitalWrite(porta,LOW);
87   delayMicroseconds(tONoff);
88 }
89 //-------------------------------------------------------------------------
90
91 void enviaSinal(int diodoSelecionado)
92 {
93    digitalWrite(atualiza, LOW); //deixa o registrador pronto
         para receber dados
94    digitalWrite(data,  LOW);
95    digitalWrite(clk,  LOW);
96    for (int i=1;i<(diodoSelecionado);i++){
97       criaPeriodo(clk);
98       verifCont++;
99    }
100   digitalWrite(data, HIGH);
101   delayMicroseconds(5);
102   digitalWrite(clk, HIGH);
```

```
103     delayMicroseconds(tONoff);
104     digitalWrite(data, LOW);
105     delayMicroseconds(5);
106     digitalWrite(clk, LOW);
107     delayMicroseconds(tONoff);
108     for (int i=0;i<(tamREG-diodoSelecionado);i++){
109         criaPeriodo(clk);
110     }
111     digitalWrite(clk,  LOW);
112     //Só para atualizar a saida do 74HC595
113     digitalWrite(atualiza, HIGH);     // "Armazenar" os 8 dados
            enviados
114     digitalWrite(atualiza, LOW);      // Comando para garantir
            atualização da saída
115
116 } //
```

APÊNDICE B – Código Programa Template

Programas/template.cpp

```
1  #include <stdio.h> /* 1 */
2  #include <stdlib.h>
3  #include <visa.h>
4  #include "hp4156b.h"
5
6  void check_err(ViSession vi, ViStatus ret) { /* 6 */
7    ViInt32 inst_err;
8    ViChar err_msg[256];
9
10   if (VI_SUCCESS > ret) {
11     if (hp4156b_INSTR_ERROR_DETECTED == ret) {
12       hp4156b_error_query(vi, &inst_err, err_msg);
13       printf("Instrument␣Error:␣%ld\n␣%s\n", inst_err, err_msg);
14     }
15     else {
16       hp4156b_error_message(vi, ret, err_msg);
17       printf("Driver␣Error:␣%ld\n␣%s\n", ret, err_msg);
18     }
19   }
20 } /* 20 */
21
22 void perform_meas(ViSession vi, ViStatus ret) { /* 22 */
23                          /* insert program code */
24
25 }
26 ViStatus main() /* 26 */
27 {
28   ViStatus ret; /* 28 */
29   ViSession vi;
30   ViChar err_msg[256]; /* 30 */
31
32               /* Starting the session */
33   ret = hp4156b_init("GPIB::17::INSTR", VI_TRUE, VI_TRUE, &vi);
       /* 33 */
```

```
34   if ((ret < VI_SUCCESS) || (vi == VI_NULL)) {
35     printf("Initialization␣failure.\n␣Status␣code:␣%d.\n", ret);
36     if (vi != VI_NULL) {
37       hp4156b_error_message(vi, ret, err_msg);
38       printf("Error:␣%ld\n␣%s\n", ret, err_msg);
39     }
40     exit(ret);
41   } /* 41 */
42   ret = hp4156b_reset(vi); /* resets 4155/4156 43 */
43   ret = hp4156b_timeOut(vi, 60000); /* sets 60 second timeout */
44   ret = hp4156b_errorQueryDetect(vi, VI_TRUE); /* turns on error
       detection */
45   perform_meas(vi, ret); /* calls perform_meas subprogram 47 */
46                /* ret = hp4156b_cmd(vi, "aa"); sends an invalid
                    command */
47                /* check_err(vi, ret); checks check_err
                    subprogram operation */
48                /* Closing the session
49                ret = hp4156b_close(vi); 52 */
50   check_err(vi, ret);
51   return VI_SUCCESS; /* 55 */
52 }
```

APÊNDICE C – Código Preliminar
Programa de Medição

Programas/measurement1.cpp

```
1  #include <stdio.h> /* 1 */
2  #include <stdlib.h>
3  #include <visa.h>
4  #include "hp4156b.h"
5
6  #pragma warning(disable:4996)
7
8  void check_err(ViSession vi, ViStatus ret) { /* 6 */
9    ViInt32 inst_err;
10   ViChar err_msg[256];
11
12   if (VI_SUCCESS > ret) {
13     if (hp4156b_INSTR_ERROR_DETECTED == ret) {
14       hp4156b_error_query(vi, &inst_err, err_msg);
15       printf("Instrument␣Error:␣%ld\n␣%s\n", inst_err, err_msg);
16     }
17     else {
18       hp4156b_error_message(vi, ret, err_msg);
19       printf("Driver␣Error:␣%ld\n␣%s\n", ret, err_msg);
20     }
21   }
22 } /* 20 */
23
24   ViInt32 drain = 1; /* SMU1 */ /* 4 */
25   ViInt32 gate = 2; /* SMU2 */
26   ViInt32 source = 3; /* SMU3 */
27   ViInt32 bulk = 4; /* SMU4 */
28
29   ViReal64 vd = 3;
30   ViReal64 vg = 3;
31   ViReal64 idcomp = 0.05;
32   ViReal64 igcomp = 0.01;
33   ViReal64 hold = 0;
34   ViReal64 delay = 0;
```

```
35  ViReal64 s_delay = 0;
36  ViReal64 p_comp = 0;
37
38  ViInt32 nop1 = 11;
39  ViInt32 nop2 = 3;
40
41  ViInt32 rep;
42  ViReal64 sc[33];
43  ViReal64 md[33];
44  ViInt32 st[33];
45
46  ViReal64 dvg[3];
47
48  ViInt32 i = 0;
49  ViInt32 j;
50  ViInt32 n;
51
52  ViChar f_name[] = "C:\Agilent\data\data1.txt";
53  ViChar head1[] = "Vg␣(V),␣Vd␣(V),␣Id␣(mA),␣Status";
54  ViChar msg1[] = "Saving␣data...";
55  ViChar msg2[] = "Data␣save␣completed.";
56  ViChar c = '/n'; /* 36 */
57
58  ret = hp4156b_setSwitch(vi, drain, 1); /* 38 */
59  ret = hp4156b_setSwitch(vi, gate, 1);
60  ret = hp4156b_setSwitch(vi, source, 1);
61  ret = hp4156b_setSwitch(vi, bulk, 1);
62  check_err(vi, ret); /* 42 */
63
64            /* 44 */
65  ret = hp4156b_force(vi, bulk, hp4156b_VF_MODE, 0, 0, 0.1, 0);
66  ret = hp4156b_force(vi, source, hp4156b_VF_MODE, 0, 0, 0.1, 0);
67  for (j = 0; j < nop2; j++) { /* 48 */
68    dvg[j] = (j + 1) * vg / nop2;
69    ret = hp4156b_force(vi, gate, hp4156b_VF_MODE, 0, dvg[j],
        igcomp, 0);
70    ret = hp4156b_setIv(vi, drain, hp4156b_SWP_VF_SGLLIN, 0, 0,
        vd, nop1, hold,
71      delay, s_delay, idcomp, p_comp);
72    check_err(vi, ret);
73    ret = hp4156b_sweepIv(vi, drain, hp4156b_IM_MODE, 0, &rep,
        &sc[i], &md[i], &st[i]);
```

```
74      check_err(vi, ret);
75
76      if (rep = nop1) {
77        i = i + nop1;
78      }
79
80      else {
81        printf("%d␣measurement␣steps␣were␣returned.\nIt␣must␣be␣%d␣
            steps.\n", rep, nop1);
82        ret = hp4156b_zeroOutput(vi, hp4156b_CH_ALL);
83        ret = hp4156b_setSwitch(vi, hp4156b_CH_ALL, 0);
84        check_err(vi, ret);
85        exit(ret);
86      }
87    } /* 67 */
88
89    ret = hp4156b_zeroOutput(vi, hp4156b_CH_ALL); /* 69 */
90    check_err(vi, ret);
91
92    printf("␣Vg␣(V),␣Vd␣(V),␣Id␣(mA)\n"); /* 72 */
93
94    for (j = 0; j < nop2; j++) {
95      n = j * nop1;
96      for (i = n; i < n + nop1; i++) {
97        printf("␣%4.2f,␣%4.2f,␣%9.6f␣\n", dvg[j], sc[i], md[i] *
            1000);
98      }
99    } /* 79 */
100
101   FILE *stream; /* 81 */
102   if ((stream = fopen(f_name, "w+")) == NULL) {
103     printf("Data␣file␣was␣not␣opened\n");
104   }
105   else {
106     printf("%s%c", msg1, c);
107     fprintf(stream, "%s%c", head1, c);
108     for (j = 0; j < nop2; j++) {
109       n = j * nop1;
110       for (i = n; i < n + nop1; i++) {
111         fprintf(stream, "%4.2f,␣%4.2f,␣%9.6f,␣%d\n", dvg[j],
              sc[i], md[i] * 1000, st[i]);
112       }
```

```
113     }
114
115     printf("%s%c", msg2, c);
116   }
117
118   if (fclose(stream)) {
119     printf("Data file was not closed\n");
120   } /* 100 */
121
122   ret = hp4156b_setSwitch(vi, hp4156b_CH_ALL, 0); /* 102 */
123   check_err(vi, ret);
124 }
125
126 void perform_meas(ViSession vi, ViStatus ret) /* 1 */
127 {
128   ViInt32 anode = 1; /* SMU1 */ /* 3 */
129   ViInt32 catode = 2; /* SMU2 */
130   ViReal64 vd = 3;
131   ViReal64 vg = 3;
132   ViReal64 idcomp = 0.05;
133   ViReal64 igcomp = 0.01;
134   ViReal64 hold = 0;
135   ViReal64 delay = 0;
136   ViReal64 s_delay = 0;
137   ViReal64 pdcomp = 0;
138   ViReal64 pgcomp = 0;
139   ViInt32 nop = 11;
140   ViInt32 rep;
141   ViReal64 sc[11];
142   ViReal64 md[11];
143   ViInt32 st[11];
144   ViInt32 i;
145   ViChar f_name[] = "C:\Agilent\data\data2.txt";
146   ViChar head1[] = "Vd (V), Id (mA), Status";
147   ViChar msg1[] = "Saving data...";
148   ViChar msg2[] = "Data save completed.";
149   ViChar c = '\n'; /* 26 */
150   ret = hp4156b_setSwitch(vi, anode, 1); /* 27 */
151   ret = hp4156b_setSwitch(vi, catode, 1);
152   ret = hp4156b_setSwitch(vi, source, 1);
153   ret = hp4156b_setSwitch(vi, bulk, 1);
154   check_err(vi, ret); /* 31 */
```

```
155
156   ret = hp4156b_force(vi, bulk, hp4156b_VF_MODE, 0, 0, 0.1, 0);
         /* 33 */
157   ret = hp4156b_force(vi, source, hp4156b_VF_MODE, 0, 0, 0.1, 0);
158   ret = hp4156b_setIv(vi, gate, hp4156b_SWP_VF_SGLLIN, 0, 0, vg,
         nop, hold, delay,
159     s_delay, igcomp, pgcomp);
160   check_err(vi, ret); /* 37 */
161
162   ret = hp4156b_setSweepSync(vi, drain, hp4156b_VF_MODE, 0, 0,
         vd, idcomp, pdcomp);
163   check_err(vi, ret);
164
165   ret = hp4156b_sweepIv(vi, drain, hp4156b_IM_MODE, 0, &rep,
         &sc[0], &md[0], &st[0]);
166   check_err(vi, ret); /* 43 */
167
168   ret = hp4156b_zeroOutput(vi, hp4156b_CH_ALL); /* 45 */
169   ret = hp4156b_setSwitch(vi, hp4156b_CH_ALL, 0);
170   check_err(vi, ret); /* 47 */
171
172   if (rep != nop) { /* 49 */
173     printf("%d␣measurement␣steps␣were␣returned.\nIt␣must␣be␣%d␣
           steps.\n", rep, nop);
174     exit(ret);
175   } /* 52 */
176
177   printf("␣Vg␣(V),␣Id␣(mA)\n"); /* 54 */
178   for (i = 0; i < nop; i++) {
179     printf("␣%4.2f,␣%9.6f␣\n", sc[i], md[i] * 1000);
180   } /* 57 */
181
182     FILE *stream; /* 59 */
183
184   if ((stream = fopen(f_name, "w+")) == NULL) {
185     printf("Data␣file␣was␣not␣opened\n");
186   }
187   else {
188     printf("%s%c", msg1, c);
189     fprintf(stream, "%s%c", head1, c);
190     for (i = 0; i < nop; i++) {
```

```
191       fprintf(stream, "%4.2f, %9.6f, %d\n", sc[i], md[i] * 1000,
              st[i]);
192     }
193     printf("%s%c", msg2, c);
194   }
195
196   if (fclose(stream)) {
197     printf("Data file was not closed\n");
198   } /* 75 */
199 }
200
201 ViStatus main() /* 26 */
202 {
203   ViStatus ret; /* 28 */
204   ViSession vi;
205   ViChar err_msg[256]; /* 30 */
206
207               /* Starting the session */
208   ret = hp4156b_init("GPIB::17::INSTR", VI_TRUE, VI_TRUE, &vi);
        /* 33 */
209   if ((ret < VI_SUCCESS) || (vi == VI_NULL)) {
210     printf("Initialization failure.\n Status code: %d.\n", ret);
211     if (vi != VI_NULL) {
212       hp4156b_error_message(vi, ret, err_msg);
213       printf("Error: %ld\n %s\n", ret, err_msg);
214     }
215     exit(ret);
216   } /* 41 */
217   ret = hp4156b_reset(vi); /* resets 4155/4156 43 */
218   ret = hp4156b_timeOut(vi, 60000); /* sets 60 second timeout */
219   ret = hp4156b_errorQueryDetect(vi, VI_TRUE); /* turns on error
        detection */
220
221   perform_meas_stair(vi, ret); /* calls perform_meas subprogram
        47 */
222
223
224   /* ret = hp4156b_cmd(vi, "aa"); sends an invalid command */
225   /* check_err(vi, ret); checks check_err subprogram operation */
226   /* Closing the session
227   ret = hp4156b_close(vi); 52 */
228   check_err(vi, ret);
```

```
229     return VI_SUCCESS; /* 55 */
230 }
```